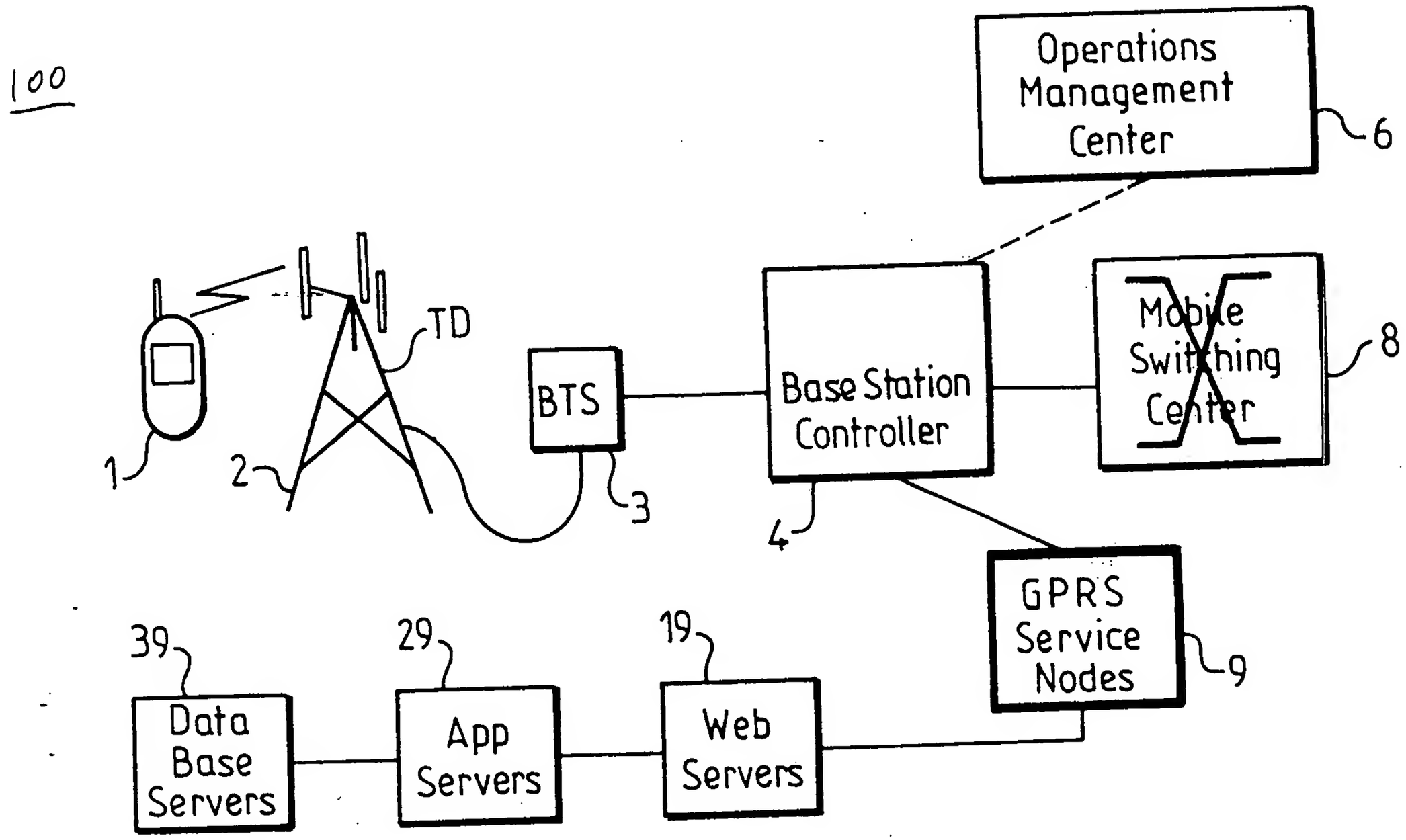
100
Operations Management Center
6
TD
BTS
Base Station Controller
Mobile Switching Center
8
1
2
3
4
GPRS Service Nodes
9
39
Data Base Servers
29
App Servers
19
Web Servers

FIG.1

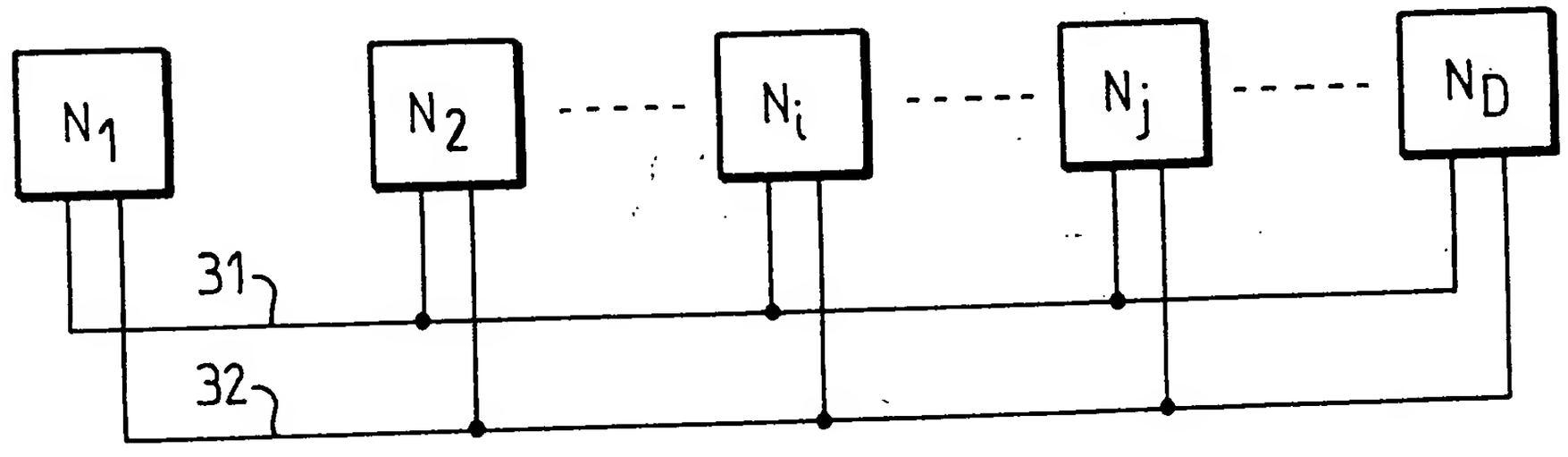
N1
N2
Ni
Nj
ND
31
32

FIG.2

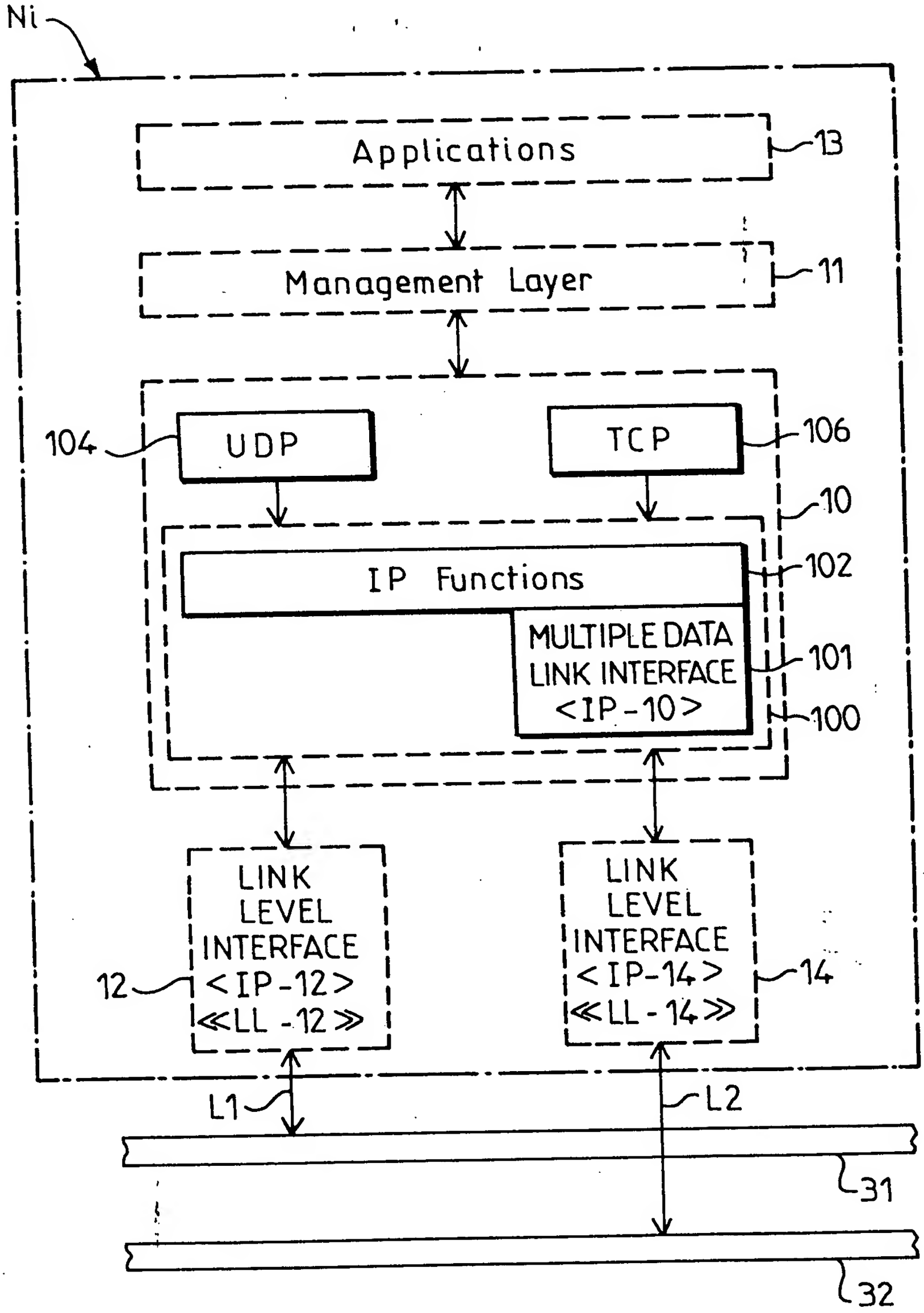
Ni
Applications
13
Management Layer
11
104
UDP
TCP
106
10
IP Functions
102
MULTIPLE DATA
LINK INTERFACE
<IP-10>
101
100
LINK
LEVEL
INTERFACE
<IP-12>
≪LL-12≫
12
LINK
LEVEL
INTERFACE
<IP-14>
≪LL-14≫
14
L1
L2
31
32

FIG.3

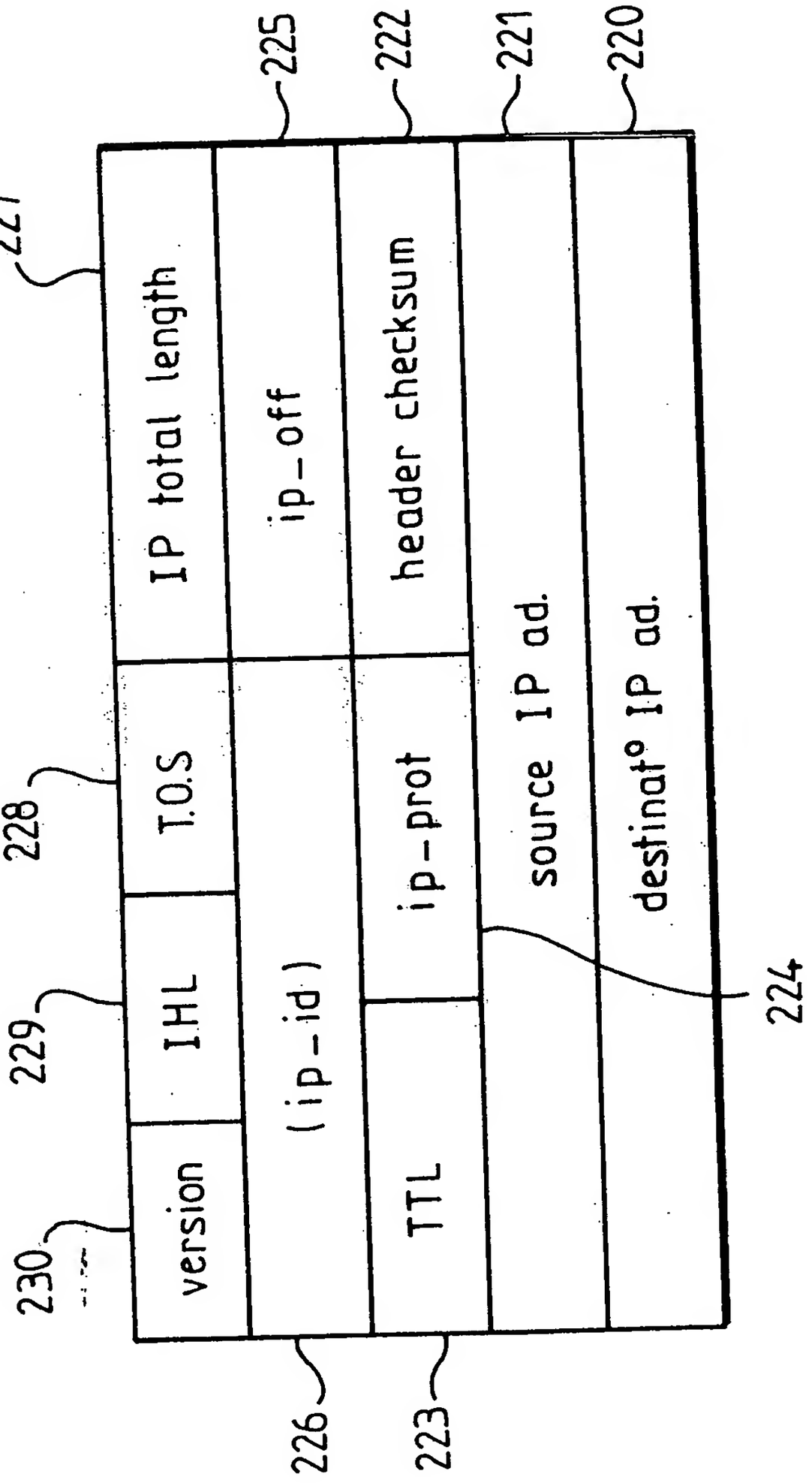
225
222
221
220
227
IP total length
ip_off
header checksum
228
T.O.S
ip-prot
source IP ad.
destinat° IP ad.
224
229
IHL
(ip-id)
TTL
230
version
226
223

FIG.4A

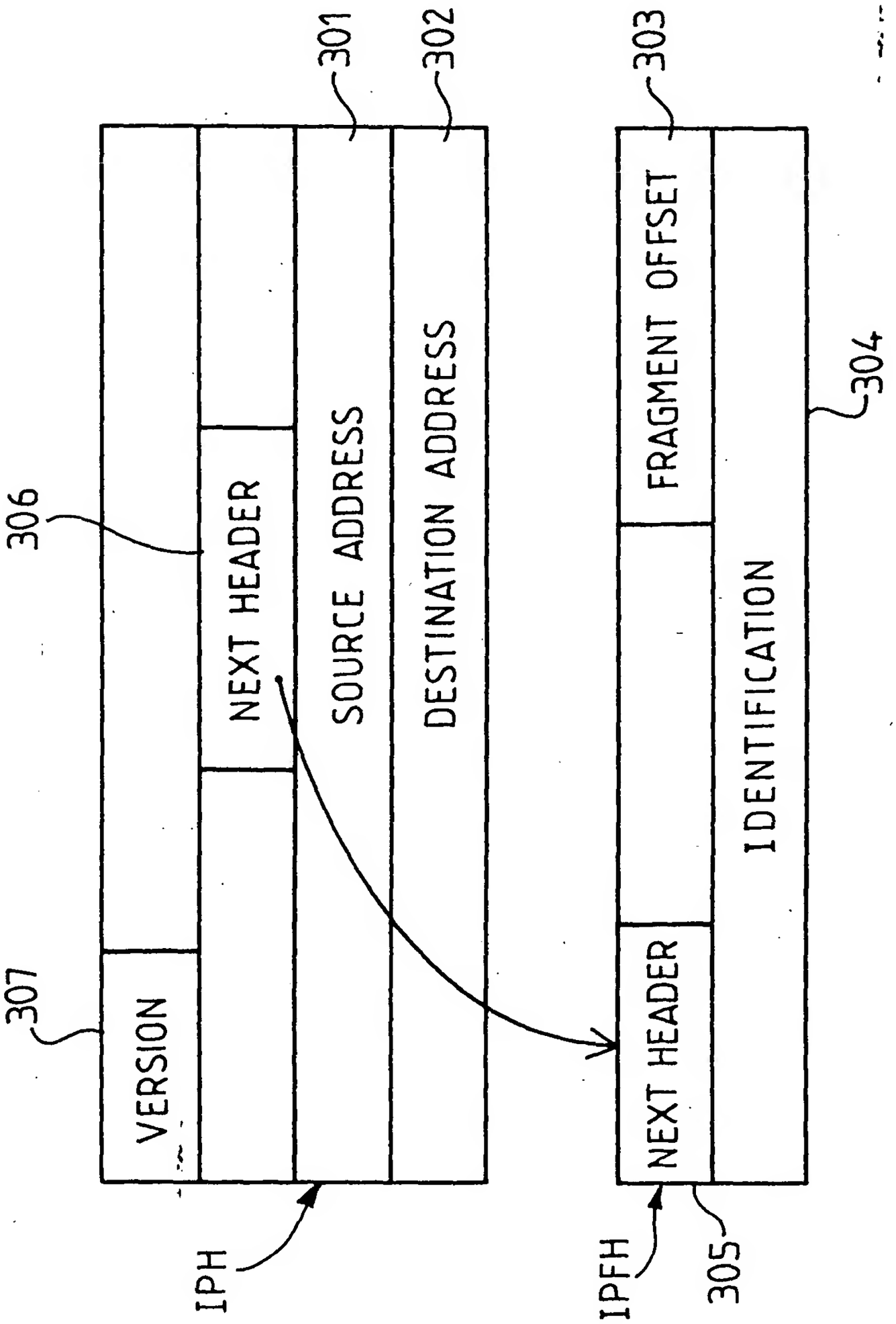
301
302
303
VERSION
NEXT HEADER
SOURCE ADDRESS
DESTINATION ADDRESS
NEXT HEADER
FRAGMENT OFFSET
IDENTIFICATION
304
306
307
IPH
IPFH
305

FIG.4B

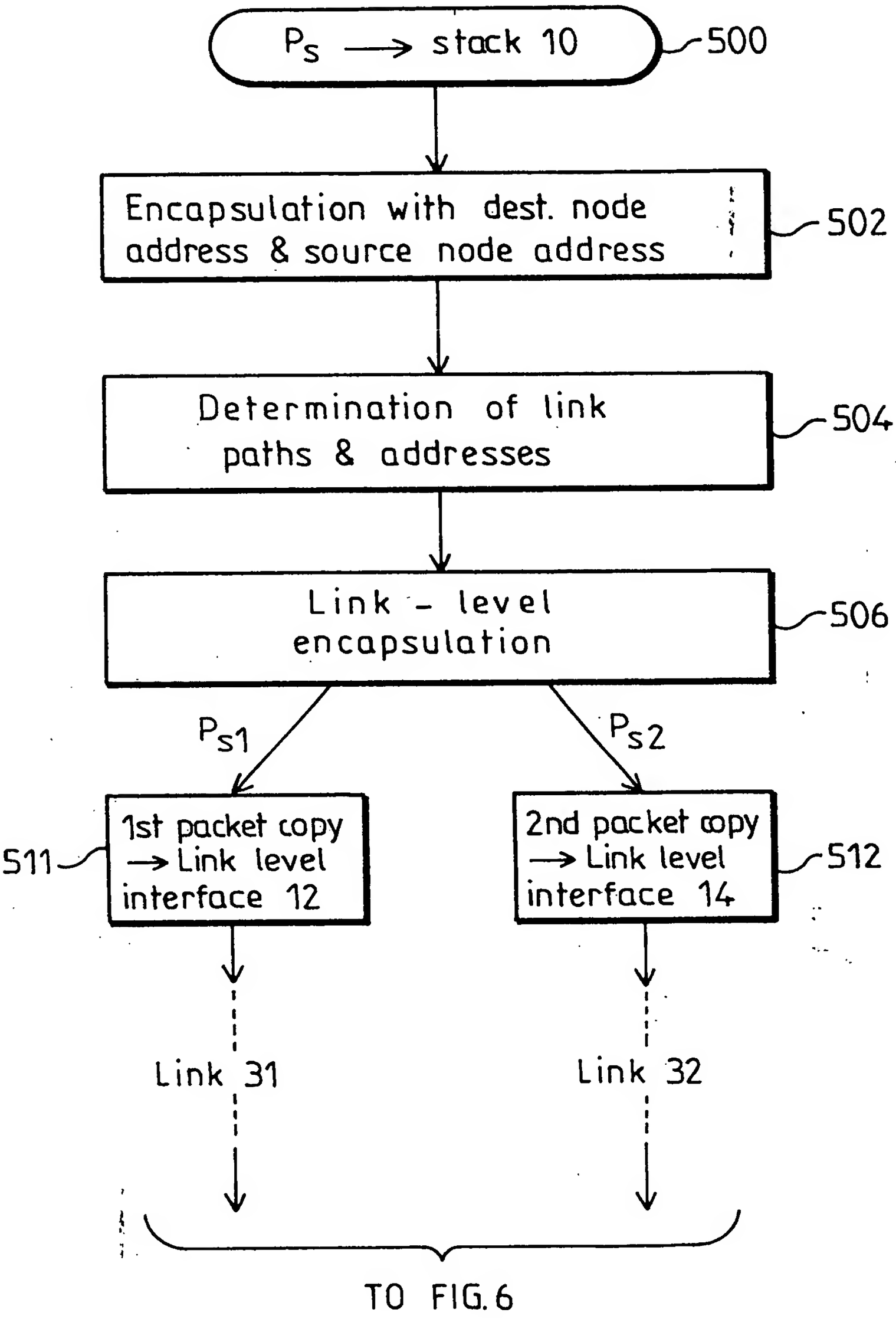
P_S → stack 10
500
Encapsulation with dest. node address & source node address
502
Determination of link paths & addresses
504
Link - level encapsulation
506
P_{S1}
P_{S2}
511
1st packet copy → Link level interface 12
2nd packet copy → Link level interface 14
512
Link 31
Link 32
TO FIG. 6

FIG. 5

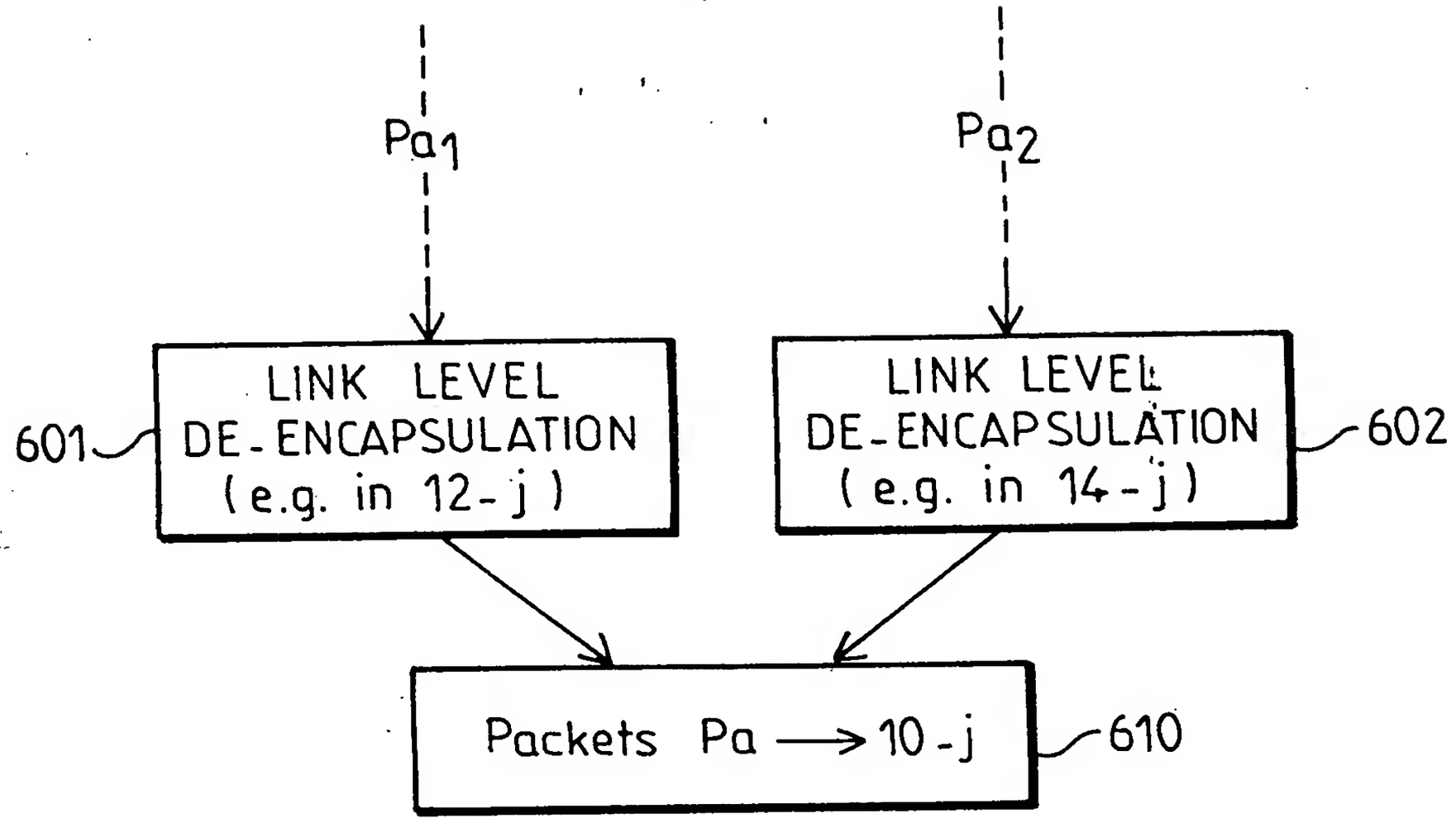
Pa1
Pa2
LINK LEVEL DE-ENCAPSULATION (e.g. in 12-j)
601
LINK LEVEL DE-ENCAPSULATION (e.g. in 14-j)
602
Packets Pa → 10-j
610

FIG.6

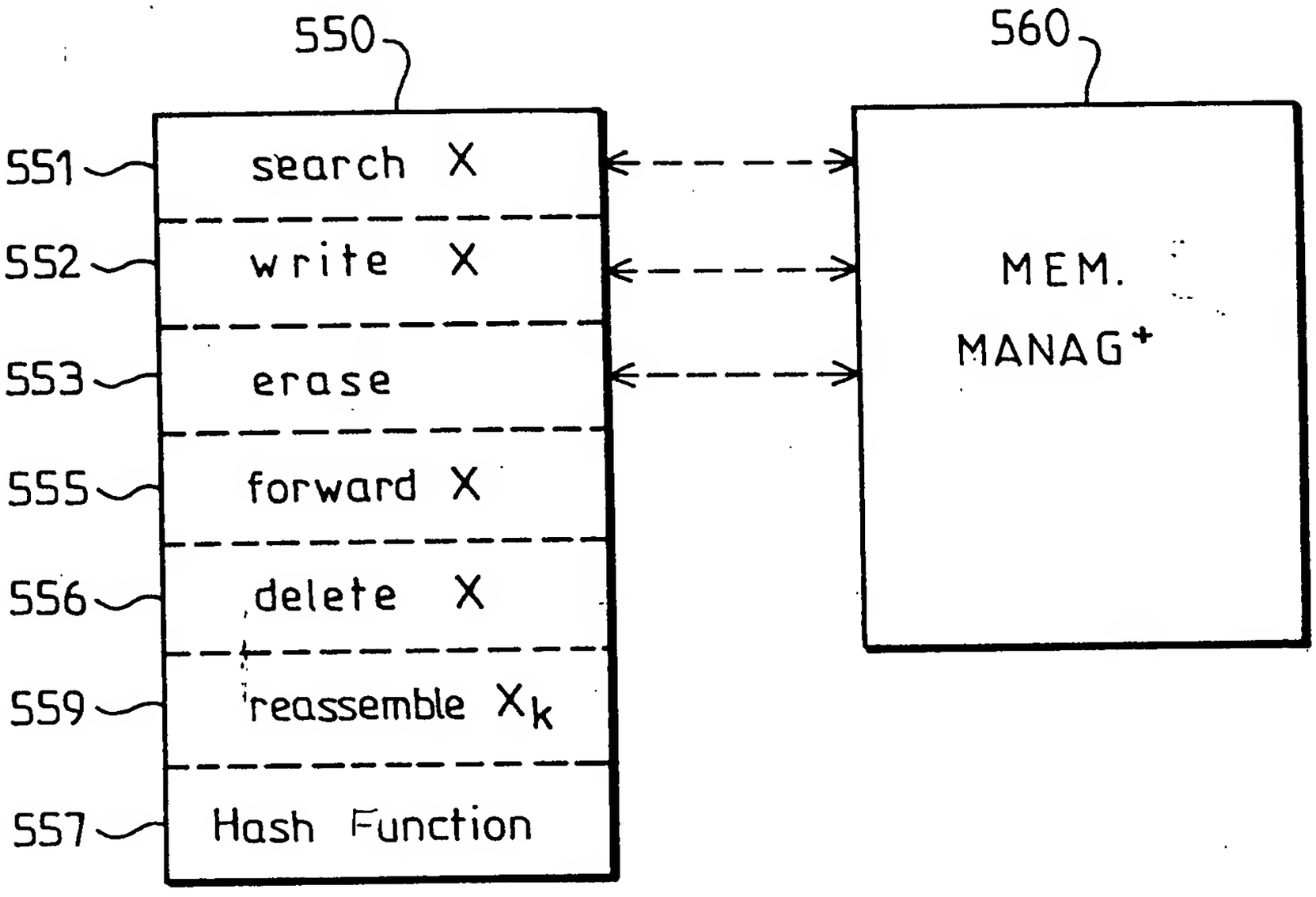
550
560
551
search X
552
write X
553
erase
555
forward X
556
delete X
559
reassemble X_k
557
Hash Function
MEM. MANAG+

FIG.7

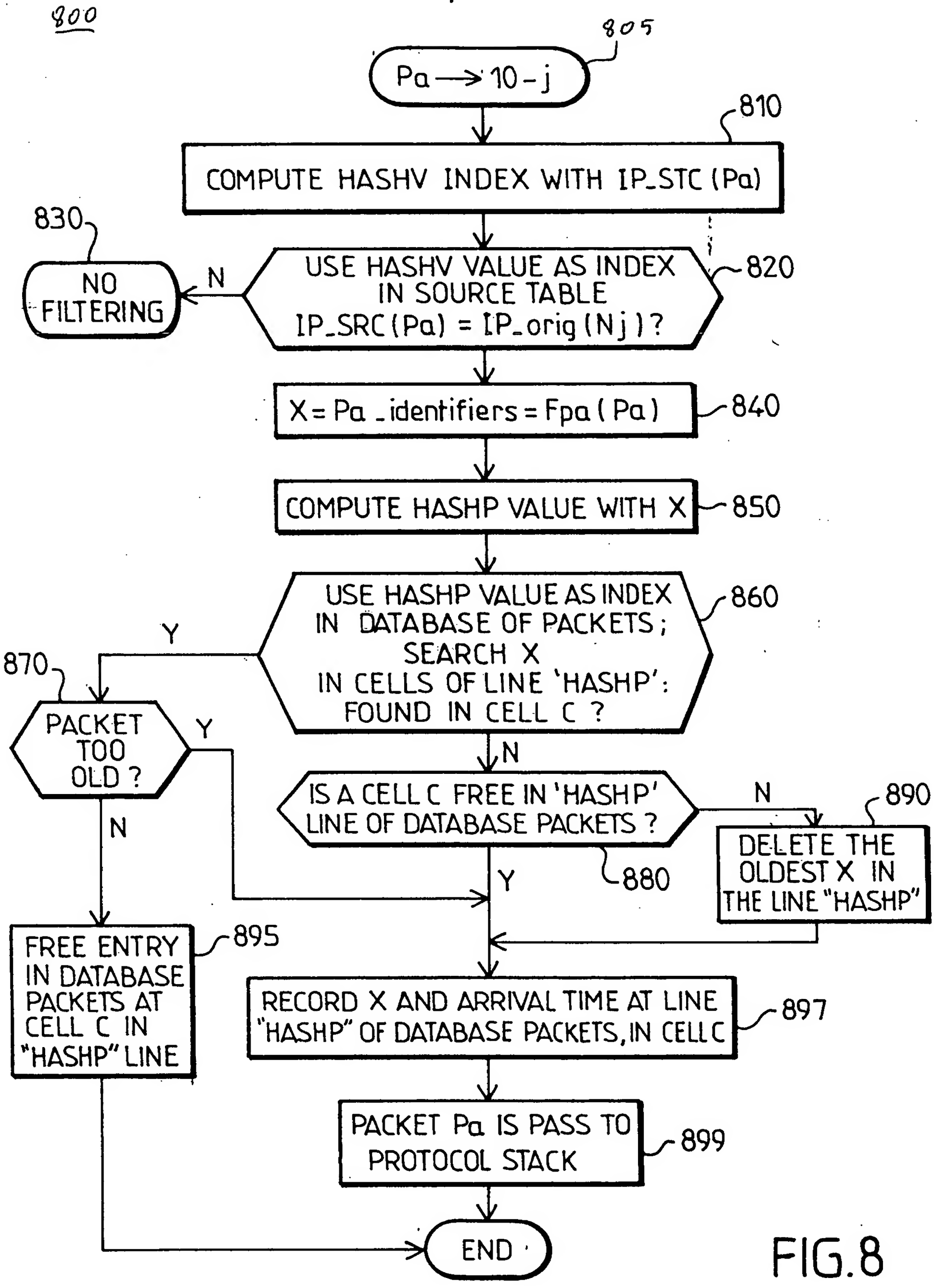
800
805
Pa → 10 - j
810
COMPUTE HASHV INDEX WITH IP_STC (Pa)
820
USE HASHV VALUE AS INDEX IN SOURCE TABLE IP_SRC (Pa) = IP_orig (Nj) ?
N
830
NO FILTERING
840
X = Pa _identifiers = Fpa (Pa)
850
COMPUTE HASHP VALUE WITH X
860
USE HASHP VALUE AS INDEX IN DATABASE OF PACKETS ; SEARCH X IN CELLS OF LINE 'HASHP': FOUND IN CELL C ?
Y
870
PACKET TOO OLD ?
Y
N
IS A CELL C FREE IN 'HASHP' LINE OF DATABASE PACKETS ?
N
890
DELETE THE OLDEST X IN THE LINE "HASHP"
880
Y
895
FREE ENTRY IN DATABASE PACKETS AT CELL C IN "HASHP" LINE
897
RECORD X AND ARRIVAL TIME AT LINE "HASHP" OF DATABASE PACKETS, IN CELL C
899
PACKET Pa IS PASS TO PROTOCOL STACK
END

FIG.8

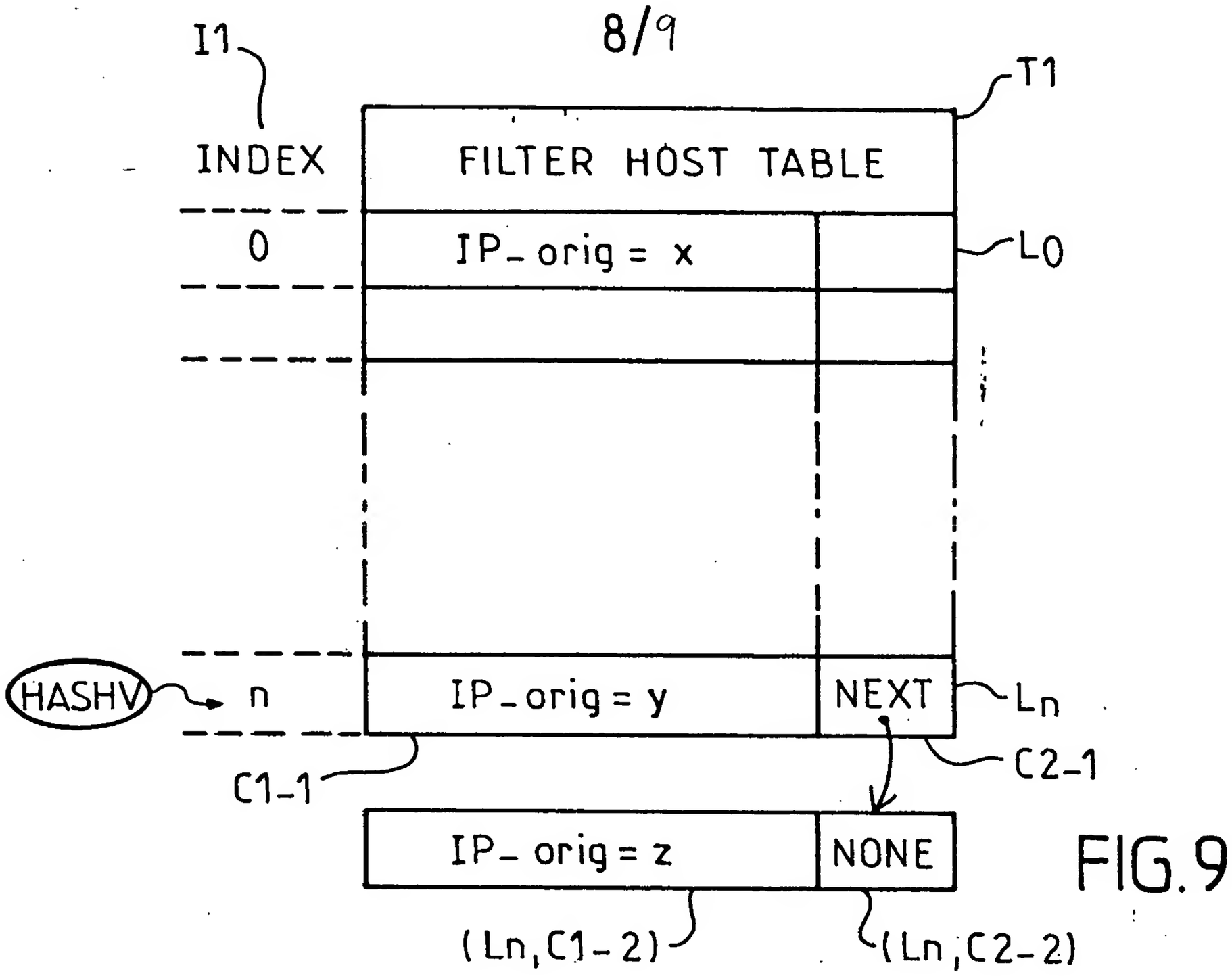
I1
INDEX
FILTER HOST TABLE
T1
0
IP_orig = x
L0
HASHV
n
IP_orig = y
NEXT
Ln
C1-1
C2-1
IP_ orig = z
NONE
(Ln, C1-2)
(Ln ; C2-2)

FIG.9

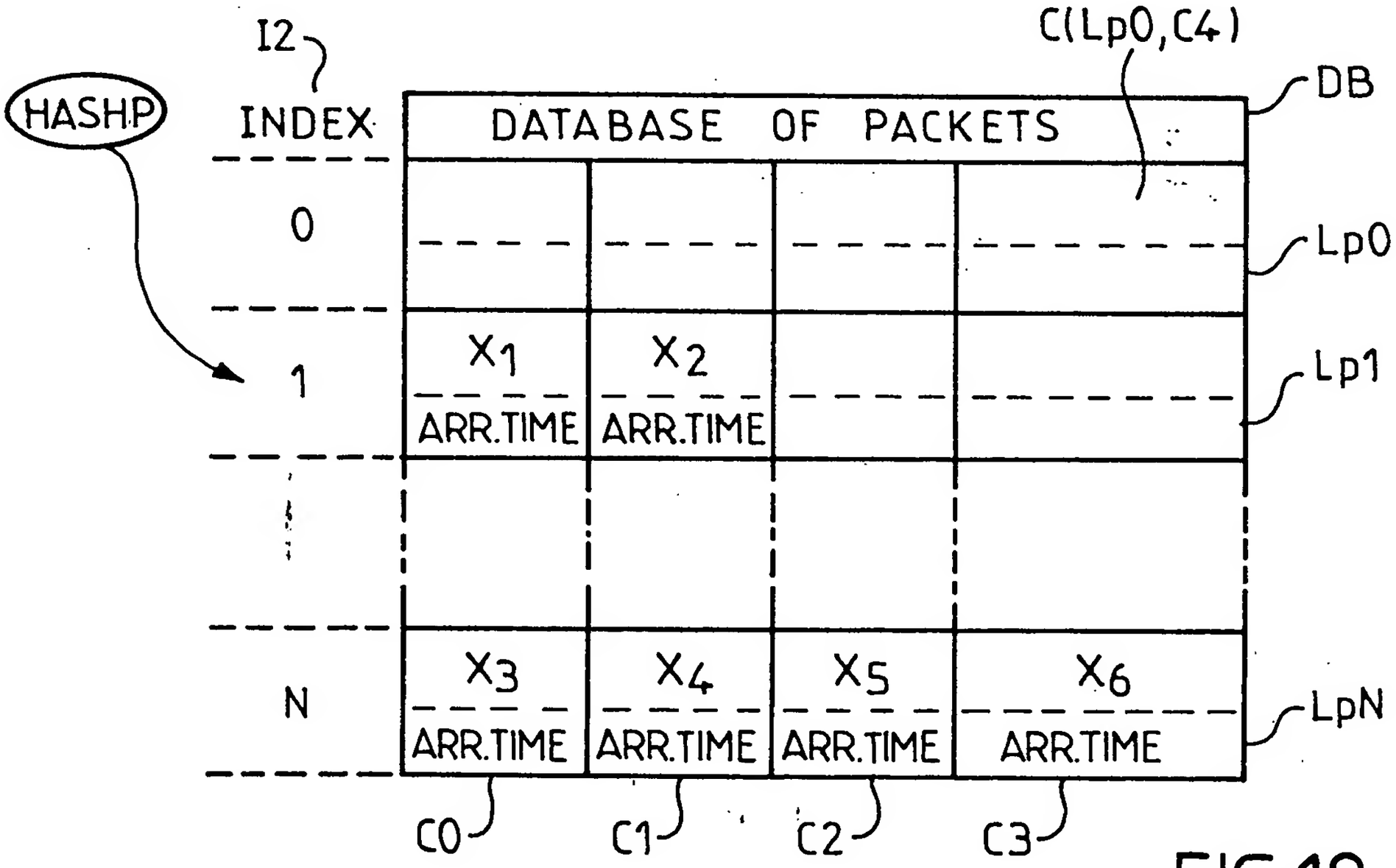
I2
HASHP
INDEX
DATABASE OF PACKETS
C(Lp0, C4)
DB
0
Lp0
1
X1
X2
ARR.TIME
ARR.TIME
Lp1
N
X3
X4
X5
X6
ARR.TIME
ARR.TIME
ARR.TIME
ARR.TIME
LpN
C0
C1
C2
C3

FIG.10

Figure 11

cgtpPpkt_footprint_t

```
1       #define CGTP_ADDRESS_NONE 0/* a free entry* /
2       #define CGTP_ADDRESS_IPV4 4/* a used IPv4 entry*/
3       #define CGTP _ADDRESS_PV6 6/* a used IPv6 entry* /
4       type of struct cgtp -addr-t { uint-t ipv; /* One of above CGTP-addresses* /
5                                   in6-addr-t addr; j* IPv6 or IPv4 mapped in IPv6*j
6                           } cgtp-addr-t;

        /* CGTP IP packet footprint * /
7       type of struct cgtp J>kt-footprint-t {
8       cgtp -addr-t addr; /* source address of incoming packet or free entry* j
9               union {
10                      union {
11                              struct {
12                                      uint8-t itf; /* incoming packet link identifier* /
13                                      uint8-t ipJ>;/*IPv4 protocol field*j
14                                      uint16-t ip-frag; /*IPv4 fragmentation field*/
15                                      uint16-t ip-crc;/* IPv4 header CRC field*j
16                                      uint16-t ip-id;/*IPv4 identification field*/
17                              } s4;
18                      }v4;
19                      union {
20                              struct {
21                                      uint8-t itf ; /*incoming packet link identifier* j
22                                      uint16-t ip6-offlg; /*IPv6 fragmentation offset*/
23                                      uint32-t ip6f-id;/*IPv6 fragment identifier* /
24                              } s6;
25                      }v6;
26              }un;
27      } cgtp J>kt-footprint-t;
```